UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

MASTEROBJECTS, INC.,

Plaintiff,

v.

AMAZON.COM, INC,

Defendant.

No. C 20-08103 WHA

ORDER RE SUMMARY JUDGMENT

INTRODUCTION

In this patent infringement action, defendant moves for summary judgment of noninfringement. To the following extent, the motion is **GRANTED**.

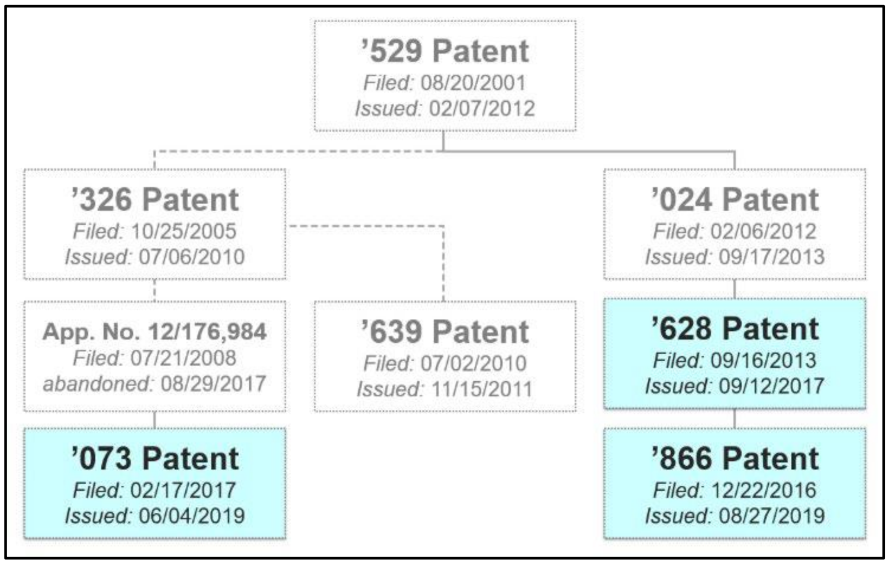
STATEMENT

This litigation concerns autocomplete technology for digital searches. Autocompletion suggests ways for the user to complete her search as she actively types it into a search bar. Patent owner MasterObjects, Inc. accuses alleged infringer Amazon.com, Inc. of infringing three of its patents in this area, U.S. Patent Nos. 9,760,628; 10,311,073; and 10,394,866.

Our parties have some history. MasterObjects first filed a patent-infringement lawsuit against Amazon in 2011, but the parties stipulated to dismissal without prejudice less than six months later. *MasterObjects, Inc. v. Amazon.com, Inc.*, No. C 11-01055 CRB (N.D. Cal. filed Mar. 7, 2011) (Judge Charles R. Breyer). MasterObjects targeted Amazon once again in May

2020, when it initiated the instant lawsuit in the United States District Court for the Southern District of New York. In October 2020, Judge P. Kevin Castel transferred the suit to our district (Dkt. No. 82). MasterObjects originally alleged infringement of four patents, but one was later withdrawn.

The three patents still in suit all descend from U.S. Patent No. 8,112,529 (filed in 2001). The following diagram lays out the patent genealogy. The asserted patents are highlighted in blue, full lines demarcate continuations of the earlier patent, and dashed lines represents continuations-in-part:



The ’628 and ’866 patents are continuations of the original ’529 patent and share its specification. The ’073 patent is a continuation-in-part of the ’529 patent and incorporates its specification by reference.

Of the fifteen claims asserted, there are three independent claims from which the rest of the claims-in-suit depend: claim 13 of the ’628 patent; claim 1 of the ’073 patent; and claim 1 of the ’866 patent. To frame the analysis that follows, here is claim 13 of the ’628 patent in its entirety (emphasis added):

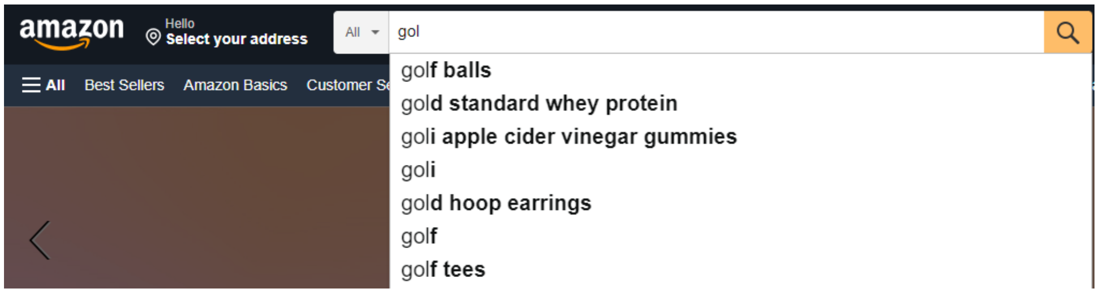
> **13[a]** A system comprising:
>
> **13[b]** a server system, including one or more computers, which is configured to receive query messages from a client object, the server system receiving and asynchronously responding to the query messages from the client object over a network;

> **13[c]** the client object that, while a user is providing input comprising a lengthening string of characters, sends query messages to the server system;
>
> **13[d]** Whereby the query messages represent the lengthening string as additional characters are being input by the user; and
>
> **13[e]** wherein the server system, while receiving said query messages, uses the input to query data available to the server system and send return messages to the client object containing results in response to the input;
>
> **13[f]** *wherein the server system caches query results and subsequently determines results by looking up the query in said cache so that it can avoid performing a query for the same input on a data source or looking up said query in a second cache.*

Per the common specification, "[a]s a user inputs data into a field on a form, the auto-complete function analyzes the developing character string and makes intelligent suggestions about the intended data being provided. These suggestions change dynamically as the user types additional characters in the string" ('628 patent 6:44–48). All agree that the claimed system generates autocomplete results from a specific server system as opposed to a broader search of the internet, generally.

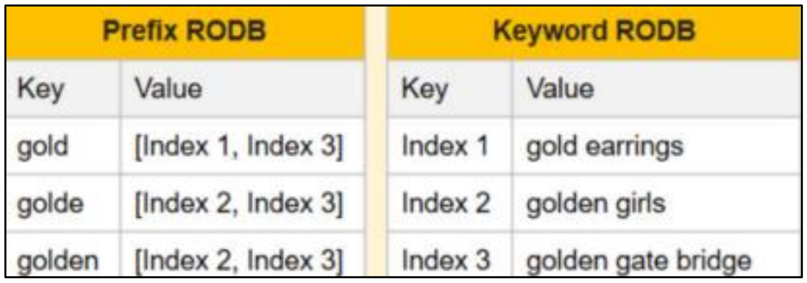
Of principal concern here, the key limitation in the claim recited above is directed to the system's use of a "cache" to provide autocomplete results. "Cache" is used as both noun and verb. All the other claims-in-suit similarly reference a cache as well as a "data source" (or "content sources"). There is no dispute that "content sources," like a "data source," generally provide data to the claimed system. The lay reader will likely have a general awareness of a cache in the computing sense, such as a web browser (*e.g.*, Mozilla Firefox) storing an iteration of a previously visited webpage (*e.g.*, cand.uscourts.gov). This order addresses the use of the term "cache" by the patents-in-suit.

Before going further into the claims, this order will get into the nuts and bolts of the accused system. Let's start with a practical example of Amazon's autocomplete as a user would find it on the Amazon.com homepage:



A user here has typed "gol" into Amazon's search bar (an autocomplete query), and Amazon's system has suggested several autocomplete query results, *i.e.*, suggestions for a complete query for products such as "golf balls," "gold standard whey protein," etc. The autocomplete query results adjust as the user continues typing and revises her autocomplete query.

Amazon's autocomplete system uses a freestanding, constructed set of databases built by a process Amazon calls the "daily build." The "daily build" process occurs every one or two days where it affirmatively builds two paired databases called read-only databases (RODBs). The RODB databases are the *only* part of Amazon's system that can process an autocomplete query, an important point for noninfringement. These RODBs are, by definition, "read only" — data can neither be added nor removed from the databases during their one-to-two-day lifespan. The paired RODBs work together and store data in key-value pairs, as follows. The *first* RODB, the "Prefix RODB," includes queries (*i.e.*, prefixes) a user might type (*e.g.*, "gold," "golde," "golden"), paired with information reflecting rows in the *second* RODB. The *second* RODB, the "Keyword RODB," stores the autocomplete suggestions (*i.e.*, keywords) that correspond to the user's query/prefix:



Prefix RODB		Keyword RODB	
Key	Value	Key	Value
gold	[Index 1, Index 3]	Index 1	gold earrings
golde	[Index 2, Index 3]	Index 2	golden girls
golden	[Index 2, Index 3]	Index 3	golden gate bridge

While a user types, the autocomplete system finds a matching prefix in the Prefix RODB, if any, then follows the index numbers to the Keyword RODB, which the system then pushes to the user (Turnbull Noninfringement Rep. ¶¶ 103, 106–12, Dkt. Nos. 380-12, 384-5; Miller Decl. Exh. F, Dkt. No. 380-14).

The key point, however, is how the columns are created in the first place. The RODBs are created from the "Tommy Query Groups." Tommy is the name for the Amazon search dataset, which includes information on searches, clicks, purchases, etc. The Tommy Query Groups are aggregated Tommy search-analytics data related to completed product searches. Tommy Query Groups contain completed search strings from product search requests and metadata associated with each search request. This bears emphasizing because the Tommy dataset does not store autocomplete queries and at most stores autocomplete results (and arguably not even that). Rather, the Tommy dataset stores completed *product search queries*, *i.e.*, not a query where the results are suggestions for a complete query (*e.g.*, "Rol" = "Rolex"), but a query where the corresponding results are listings for actual products Amazon has for sale (*e.g.*, the range of Rolex watches a customer could buy).

The "daily build" process uses data from the Tommy Query Groups to populate the Keyword RODB (the results column). During this process, among other things, spelling-corrected suggestions are generated and a blocklist is applied to remove certain entries. Once the "daily build" collects all the keywords in the Keyword RODB, it generates prefixes for each using a rote, progressive letter construction: for a keyword such as, *e.g.*, "Rolex," it would generate the prefixes of "R," "Ro," "Rol," "Role," and "Rolex." Amazon also uses a "Personalization Platform Data Service" (PPDS), which includes the individual user's recent completed *product search queries*, which are used to re-rank suggestions to more closely match those prior queries (*see* Turnbull Noninfringement Rep. ¶¶ 109–12, 171, 181 n.21, 244; Peck Rep. ¶¶ 140–41, Dkt. Nos. 380-6, 384-7, 389-2).

Having completed this high-level overview of the patents-in-suit and the accused system, this order proceeds to analyze the specifics. This order follows full briefing and oral argument.

ANALYSIS

A patent infringement analysis involves two steps. The claim must be properly construed to determine its scope and meaning. Claim terms generally take "their ordinary and customary meaning," that is "the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention." *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312–13 (Fed. Cir. 2005) (en banc). Although construction begins with the claim language itself, "the specification is the single best" — and usually dispositive — "guide to the meaning of a disputed term." *Network-1 Techs., Inc. v. Hewlett-Packard Co.*, 981 F.3d 1015, 1022 (Fed. Cir. 2020) (quoting *Phillips*, 415 F.3d at 1314–15).

The properly construed claim must then be compared to the accused device or process. To constitute infringement, an accused product must practice every limitation of a properly construed claim. *Tessera, Inc. v. Int'l Trade Comm'n*, 646 F.3d 1357, 1364 (Fed. Cir. 2011); *Carroll Touch, Inc. v. Electro Mech. Sys., Inc.*, 15 F.3d 1573, 1576 (Fed. Cir. 1993).

1. AMAZON'S SYSTEM DOES NOT HAVE A CACHE OF PRIOR AUTOCOMPLETE QUERIES, AS REQUIRED BY THE CLAIMED SYSTEM.

Amazon asserts the claims-in-suit all require a cache that stores prior autocomplete *queries*. MasterObjects opposes that the claimed system only requires prior autocomplete *results* and that Amazon's construction reads limitations from the embodiment into the claims. The crux of the dispute is the construction of "cache." Because this order construes the claimed system's "cache" to store autocomplete *queries* (as well as *results*), Amazon's system does not infringe.

All of the claims-in-suit include "cache and content source" terms. The three independent claims-in-suit recite:

- "[T]he server system caches query results and subsequently determines results by looking up the query in said cache so that it can avoid performing a query for the same input on a data source or looking up said query in a second cache" ('628 patent claim 13[f]).
- "[M]atching, by the server system, the string [representing an incomplete search query] to entries in a cache of query strings and search results based on content queries received from multiple users, whereby cached search results contain a subset of data from one or more content sources" ('073 patent claim 1[e]).

- "[M]atching, by the server system, the string [representing an incomplete version of the search query] to entries in a cache of queries and search results previously retrieved from one or more content sources" ('866 patent claim 1[e]).[1]

In their claim construction statements, Amazon construes each of these claim elements as a whole, while MasterObjects construes various component terms. For example, here are the proposed constructions for claim 13[f] of the '628 patent (the proposed constructions for the other two cache limitations are similar):

MasterObjects	Amazon
• "Caches query results:" saves query results through a cache; • "Cache:" a memory store; • "Data Source:" an underlying data source; • The remainder: plain and ordinary meaning; and • This term is not limited to a system that can, if no matching cache entry is found, query another content source	"the server system saves a copy of the query result set retrieved in response to the query message and determines responses to later query messages by checking whether result sets responsive to those later query messages are present in its store of previously saved query result sets before querying a separate data source or a second cache" *This claim requires that the claimed system be able to obtain query result sets from sources other than the store of query result sets previously saved by the server system.*

The use of the word "cache" by itself begs the question of what must be stored in the cache. A cache, in common parlance, is simply a fast memory buffer for holding frequently called data (*see* Webster's Pocket Computer Dictionary, Turnbull Claim Const. Resp. Exh. 29; *see also id.* Exh. 28). What must be stored (autocomplete queries and results or just results?) is the question.

All three independent claims indicate that, a "cache," in context, is not simply "a memory store." *First*, claim 13 specifies that the system "caches query results," but goes on to explain that the system "look[s] up the query in said cache." The system's ability to look up a query in

[1] This order has labelled each individual claim limitation, *e.g.*, claim 13[f], for ease of reference.

the cache inherently indicates the system has also stored the prior query as well. Furthermore, as indicated by "subsequently" and "avoid," the use of the verb form of cache expresses that data stored in the cache had previously been processed by the data source. *Second*, claim 1[e] of the '073 patent similarly recites that the claimed system has a "cache of query strings," *i.e.*, store of autocomplete queries. The limitation describes the claimed system's cache as being "a subset of data from one or more content sources," where that data is "based on content queries *received* from multiple users." This description of the cache also evinces that it stores autocomplete queries already processed by a data source. *Third*, claim 1[e] of the '083 patent explicitly recites that the system has a "cache of queries and search results" that have been "previously retrieved from one or more content sources." This limitation makes clear the cache stores autocomplete queries previously processed by a content/data source.

Upon review, this order construes a "cache" as a particular type of memory store that saves a copy of both the autocomplete query and results from the data source. This order likewise construes the verb form of "cache" to mean to store a copy of both the autocomplete query and results from the data source.

Shifting sands have blown hard in this case. MasterObjects began this litigation expressly in agreement that the claims required a cache of both autocomplete queries and results. As this litigation progressed, however, it has become clear that the accused system does not store queries, so MasterObjects flip-flopped in its opposition to summary judgment, reneging on its prior assertions that the cache terms did indeed require storage of autocomplete queries. For example, in MasterObjects' claim construction briefing:

> Amazon begins by misdescribing MasterObjects' proposed definition of the caching terms . . . . That is, MasterObjects [sic] ***claimed cache is based on prior search queries (e.g., "ROL") with prior query results (e.g., "Rolex watches"). This is not a subtle point in MasterObjects' briefing***

(MO Claim Const. Reply Br. 5, Dkt No. 262 (emphasis added); *see also* MO Claim Const. Br. 1–2, Dkt. No. 236 (same); *id.* at 22 (same); Fourth Supp. Infringement Contentions (FSIC) at 78, Dkt. No. 388-8 (same)). MasterObjects went on, discussing its cache construction in other cases:

> But the real MasterObjects construction here **accords** with its prior construction: (1) *Meta*: "cache" is "a store that includes previous queries and search results retrieved in response to previous queries;" (2) *eBay*: "query and result cache" is "[a] cache which stores previous queries and content or other information returned in response to the previous queries;" and (3) *Yahoo!*: "query and result cache" is "[a] cache which stores previous queries and content or other information returned in response to the previous queries." These are vastly contradictory constructions?

(MO Claim Const. Reply Br. 5, emphasis in original).[2]

MasterObjects' Expert Peck stated the same conclusion multiple times in his infringement report:

- "AC [Amazon's autocomplete system] does so by using multiple pre-computed query results in "Query and Results Caches" in its server-side backend. Generally, a Query and Results Cache is a memory store of prior queries and search results" (Peck Rep. ¶ 120).
- "The 13[F] claim language describes a process where: (1) a server system caches query results, which inherently includes the prior queries themselves (as one would not know what a query result was absent having the query in question) . . . ." (*id.* at ¶ 198).
- "In the language of the specification and claims, [claim 13] requires that Amazon build a cache of prior queries and results, *e.g.*, 'ROL' likely means Rolex. . . . [T]he cache is built and designed to correlate prior queries (which can be query fragments, *e.g.*, 'ROL') with the list of most likely relevant results" (*id.* at ¶ 199).

The reader should please note this is MasterObjects' *current* expert infringement report disclosed in advance of trial. This order agrees with the original and persistent claim construction proffered by MasterObjects itself.

[2] Citing *MasterObjects, Inc. v. Meta Platforms, Inc.*, No. C 21-05428 WHA (N.D. Cal.); *MasterObjects, Inc. v. Yahoo!, Inc.*, 2013 WL 6185475 (N.D. Cal. Nov. 26, 2013) (Judge Jeffrey S. White); *MasterObjects, Inc. v. eBay, Inc.*, 2013 WL 1287428 (N.D. Cal. Mar. 28, 2013) (Judge Jacqueline Scott Corley). *See also MasterObjects, Inc. v. Google, Inc.*, 2013 WL 2319087 (N.D. Cal. May 28, 2013) (Judge Phyllis J. Hamilton).

Other intrinsic evidence supports this construction, such as the *inter-partes* review proceeding for the '073 patent. In that proceeding, MasterObjects' Expert Dr. Michael J. Pazzani opined regarding claim 1[e]:

> Even if the predictive search server [of the prior art] stores that information that does not mean it is necessarily stored in a cache. A POSITA would understand that all data is stored in memory to make it usable by system [sic]. Merely storing data in memory (without more) does not require or even suggest to a POSITA that the data is stored in a cache, at least because the data stored must include "previous queries and search results," and those results must have been "retrieved in response to previous queries"

(Amazon Claim Const. Resp. Exh. 35, Dkt. No. 252-57). Dr. Pazzani expressly states the claim element requires storage of previous queries.

The common specification also accords with the construction this order has adopted. MasterObjects now decries analysis of the specification and asserts that Amazon "commits perhaps **the** cardinal sin of construction errors — importing embodiment detail into the claim language." The common specification describes one preferred embodiment of the invention, "QuestObjects." MasterObjects says the embodiment is "replete with odd neologisms" that do not appear in the claims (MO Claim Const. Reply Br. 1, emphasis in original).

This order cannot, however, simply ignore the specification — the specification "is always highly relevant to the claim construction analysis." *Phillips*, 415 F.3d at 1315 (citation omitted). The common specification states: "to ensure scalability, systems built around the present invention can be divided into multiple tiers, each tier being capable of caching *data input* and output" ('628 patent 6:16–18, emphasis added). This description is discretionary ("can"), but it provides insight into how to read the cache limitations. Furthermore, this statement in the specification is made before the introduction of the QuestObjects embodiment (*id.* at 7:24–32). Accordingly, the reference to "the present invention" indicates the description refers to the invention as a whole rather than only the preferred embodiment. *See Verizon Servs. Corp. v. Vonage Holdings Corp.*, 503 F.3d 1295, 1308 (Fed. Cir. 2007).

The QuestObjects "Glossary" also defines a "Content-based Cache" as a "persistent store of Queries and corresponding Result Sets executed by a Content Engine for a specific Content

Channel" (*id.* at 10:36–38). (A "Content Engine," per QuestObjects, is a data source, while a "Content Channel" provides one type of information from one Content Engine (*id.* at 10:39–45).)

Yes, QuestObjects is an embodiment. But this order merely takes note of the fact that the specification accords with the other intrinsic evidence regarding the cache limitations. Take, for example, Judge Phyllis Hamilton's claim construction order in MasterObjects' prior litigation against Google. There, Google pointed to the same "Glossary" definition cited above and tried to incorporate the word "persistent" into the construction of "content-based cache" and "query and result cache." Judge Hamilton noted MasterObjects' inconsistency by distancing itself from the "Glossary" definition, but found including "persistent" would create more ambiguity, not less. *See Google*, 2013 WL 2319087, at *4. This order cites the "Glossary" definition to highlight the uniform description of the cache limitations in the intrinsic evidence. "Where, as here, a patent repeatedly and consistently characterizes a claim term in a particular way, it is proper to construe the claim term in accordance with that characterization." *Wis. Alumni Research Found. v. Apple Inc.*, 905 F.3d 1341, 1351 (Fed. Cir. 2018) (citations and quotations omitted); *see also Decisioning.com, Inc. v. Federated Dep't Stores, Inc.*, 527 F.3d 1300, 1307–08, 1311 (Fed. Cir. 2008); *Phillips*, 415 F.3d at 1323.

MasterObjects now tries to walk back its prior statements with the excuse that its "prior vernacular has not always been perfect" (Opp. 16). But its citations to other passages in Expert Peck's reports to support its current position are either inapposite descriptions of Amazon's system or general statements that are consistent with Expert Peck's specific statements highlighted above (*ibid.*, citing Peck Rep. ¶¶ 153–56; Peck Invalidity Rebuttal Rep. ¶¶ 44, 46, 50, Dkt. No. 383-6).

MasterObjects now primarily relies on its proposed plain-meaning constructions for various parts of the cache limitations to argue that it has consistently asserted that only prior autocomplete *results* are required to practice the claims-in-suit. Claim 1[e] of the '866 patent for example, describes in relevant part "a cache of queries and search results previously retrieved from one or more content sources." MasterObjects argues the phrase "previously retrieved from one or more content sources" only modifies "search results," not "queries" (Opp. 14–15). So, according to MasterObjects, the plain language does not require the cache to store prior autocomplete queries. But reaching this conclusion reads "cache" out of the claims and untethers the query results from the query itself. Construed in the proper context, a "*cache* of queries and search results" indicates that the cache stores an autocomplete query along with the corresponding results generated from processing of that query. MasterObjects' argument that autocomplete queries come from the user rather than the backend server is a strawman. The point is that autocomplete queries that the system processes to generate results are stored in the cache because (in the words of MasterObjects' expert), "one would not know what a query result was absent having the query in question" (Peck Rep. ¶ 198). *See Irdeto Access, Inc. v. Echostar Satellite Corp.*, 383 F.3d 1295, 1300 (Fed. Cir. 2004).

MasterObjects' "best" argument is as follows: even though the autocomplete query is not stored in a "cache," the accused system generates a sequence of prefixes — one of which by definition must correspond exactly to the query — and stores them in a "cache." More specifically, for example, compare these: under the patented system, a user types the autocomplete query "Rol" and one of the query results will be "Rolex." It then stores both "Rol" and "Rolex" in respective columns in its cache. Pursuant to this argument, under Amazon's system a user types the autocomplete query "Rol" and one result is "Rolex." "Rolex" is then stored in a "cache." (More on what is actually stored in a moment.) The accused system then uses a progressive letter construction to store the following: "R," "Ro," "Rol," "Role," and "Rolex." All of these are stored in the "cache" as well. The main problem with this argument is the autocomplete query itself is not stored and the prefix column in the "cache" is not a history of prior queries. Rather, the Prefix RODB is an artificially generated set of all possible prefixes that would lead to "Rolex." "Rol" in the accused system is not a copy of the autocomplete query but an independently and artificially generated construct. This is sufficient for a finding of noninfringement.[3]

[3] Amazon's system could also hypothetically have the prefix "Role" for the autocomplete result "Rolex" even though no prior user had ever actually typed in the autocomplete search "Role,"

Furthermore, setting aside that the claims in-suit require caching autocomplete queries, Amazon's system also does not infringe because the Keyword RODB is built from prior *product search queries*, not prior autocomplete results like the claimed system. MasterObjects argues that a *product search query* selected by a user from an autocomplete result *is* an *autocomplete result* because the query run comes verbatim from the autocomplete result, *e.g.*, both would be "Rolex" (Opp. 5). The problem with this theory is that Amazon processes product search queries with an entirely different search system located on its own separate servers (Turnbull Noninfringement Rep. ¶¶ 125–26). Amazon's system may use information gleaned from a user's prior actions to help generate autocomplete results, but it gathers that data from an entirely different search system (for product searches). And MasterObjects' claimed system specifically caches *autocomplete* data. That is not the same. The same reasoning applies for why the PPDS does not qualify as a cache. (Regardless, the PPDS also cannot be queried and does not otherwise push autocomplete results to the user; it is only used to re-rank an individual user's autocomplete results.) Consequently, there is no infringement on this basis as well.

2. Amazon's System Does Not Have An Underlying Data Source That Can Itself Be Queried, As Required By The Claimed System.

The reason that MasterObjects painted itself into a corner regarding the claimed system's caching of prior autocomplete queries has to do with the parties' dispute over the overarching structure of the claimed system. As explained, a "cache" is a particular type of memory store that saves a copy of both the autocomplete query and results from the data source. This inherently requires the claimed system to be able to query the data source (or "content sources") for autocomplete query results. Because the "Tommy Query Groups" dataset that the accused system uses to build the RODBs cannot itself be queried for autocomplete results, Amazon does not infringe.

which could not happen for MasterObjects' claimed system.

The analysis of the claim language above indicated not only that all three cache limitations required caching of prior autocomplete queries, but also that the claimed system has an underlying data source that itself can be queried. As just one illustration to reorient the reader, claim 13[f] of the ’628 patent recites a system that “*can avoid* performing a query for the same input on a data source.” This permissive language indicates the data source itself can be queried if need be.

The common specification accords with this construction. It says, prior to the introduction of the QuestObjects embodiment: “Using the present invention, the suggestions generated by the server may, at the option of the application developer, be cached on the middle tier or on the client itself to maximize performance” (’628 patent 7:3–6). Since caching is a discretionary optimization, the claimed system must be able to query the underlying data source to generate autocomplete query results. *See Verizon*, 503 F.3d at 1308.

This order also notes that the rest of the specification uniformly and repeatedly describes the data source as a part of the system that itself can be queried (*e.g.*, ’628 patent at Figs. 2, 7a, Cols. 10:36–38; 15:31–35; 16:33–37; 21:31–34). *See Wis. Alumni Research Found.*, 905 F.3d at 1351; *see also Trustees of Columbia Univ. v. Symantec Corp.*, 811 F.3d 1359, 1365 (Fed. Cir. 2016); *Decisioning.com*, 527 F.3d at 1307–08, 1311.

In opposing this construction of the cache limitations, MasterObjects proffers an ultimately unpersuasive theory regarding a “cache miss.” It explains:

> Essentially, Amazon argues that since a cache is described as a way of scaling an instant search system, the term cache somehow must include the embedded notion of hitting a second data source in the event of a cache miss. Why? Some claims require this; others explicitly do not. *Compare* Ex. 6 (’326, not asserted), Claim 1, *with* Ex. A (’628, asserted), Claim 13. Claim differentiation alone proves Amazon wrong

(MO Claim Const. Reply Br. 6). Here are the relevant claim elements MasterObjects cites:

- “[A server that] automatically matches the increasingly focused query string both initially by matching the query string against the previously determined results stored in the unified query cache at the server, and subsequently, if no matching cache entry was found, by matching the query string against the content sources as retrieved by the server" (U.S. Patent No. 7,752,326 claim 1[d]).

- "[T]he server system caches query results and subsequently determines results by looking up the query in said cache so that it can avoid performing a query for the same input on a data source or looking up said query in a second cache" ('628 patent claim 13[f]).

MasterObjects cites claim 13 of the '628 patent as a representative example of a claim that does not require "hitting" a data source on a "cache miss."

MasterObjects' argument starts from a false premise. It does not substantively dispute the assertion that information stored in the cache comes from queries of the underlying data source. It instead addresses the orthogonal issue of whether the claimed system can return autocomplete results to a user without needing to check the data source *every time*. Indeed, claim 13 of the '628 patent specifically addresses the scenario where the "cache miss" has *already occurred* — the system then queried the data source, after which it cached that information for future autocomplete queries. Claim 13[f] contemplates that the data source is the primary underlying structure that processes autocomplete queries and generates corresponding results. As Amazon's expert, Dr. Turnbull, explains, caching "is saving a copy of something a computing system has already done or accessed during normal operation of the system . . . . [A] cache is generally understood as a temporary storage location that is an alternative to — but does not replace — calls to a different, more permanent, storage location" (Turnbull Claim Const. Decl. ¶ 105, Dkt. No. 252-21).

Indeed, MasterObjects has difficulty avoiding the same conclusion: "The purpose of the cache is to accelerate processing by *avoiding repeated organic searches* for each query" (MO Claim Const. Br. 22, emphasis added). The system's "data source" constitutes the normal, underlying source for processing those "organic" autocomplete queries and returning autocomplete results. Since the claimed system's cache saves a copy of the autocomplete query and the result generated by the data source, the data source itself can inherently be queried.

Because Amazon's system does not practice the claimed cache-and-data-source structure recited by the claims-in-suit, there can be no infringement. MasterObjects has accused the RODBs to be the cache in Amazon's system, which in turn requires the Tommy Query Groups to constitute the system's data source. As explained, the Tommy Query Groups dataset cannot itself be queried for autocomplete results. (Nor can the PPDS or the raw data maintained in Tommy itself be queried for autocomplete results for that matter.) MasterObjects argues "a cache can be a content source" (Opp. 18). This argument is a nonstarter because a cache cannot be a content source for itself under the language of the claims. Something must "underlie" the cache that itself can be queried. With no such structure, there is no infringement.

Furthermore, in light of this order's construction of the cache limitations, MasterObjects cannot demonstrate infringement at trial. MasterObjects' Expert Peck did not offer an opinion under Amazon's proposed construction and hence did not identify an underlying data source for the "cache" that can itself be queried (Peck Dep. 42–43, Dkt. No. 380-8; Turnbull Noninfringement Rep. ¶¶ 167 n.17, 208–13, 227–28). The foregoing analysis demonstrates that this patent dispute involves complex technology; Amazon has also put up its own expert negating infringement. MasterObjects, in turn, has stated that it would only use retained experts to prove infringement (Miller Exh. J). Consequently, with no relevant expert testimony, this order finds that MasterObjects cannot prove infringement. *See Centricut, LLC v. Esab Grp., Inc.*, 390 F.3d 1361, 1369–70 (Fed. Cir. 2004). Moreover, the time has also long passed for MasterObjects to revise its infringement contentions to capture this order's construction, which adopts Amazon's proposal regarding underlying data sources (*see e.g.*, FSIC at 10, 78). *See generally Fluidigm Corp. v. IONpath, Inc.*, 2020 WL 5073938 (N.D. Cal. Aug. 25, 2020). Because of this order's other conclusions, it need not reach the question of whether Mr. Peck should be excluded.

CONCLUSION

With the help of our Special Master Harold J. McElhinny, the Court has weathered a barrage of discovery disputes and motion practice in this action. The Court thanks Special Master McElhinny for his excellent work.

For the reasons stated, Amazon's motion for summary judgment of noninfringement is **GRANTED**. The foregoing disposes of literal infringement of all claims-in-suit. The summary judgment motion did not move on doctrine of equivalents. Therefore, pursuant to Rule 56(f), within **FIVE CALENDAR DAYS**, each side shall submit a statement up to **FIVE PAGES** in length concerning the proper course for dealing with the doctrine of equivalents. After reading the five-page statements, the Court may possibly vacate the trial date. Meanwhile, however, the pretrial conference and trial date stand.

IT IS SO ORDERED.

Dated: September 15, 2022.

[illegible]

WILLIAM ALSUP
UNITED STATES DISTRICT JUDGE